Atos Origin

Generic Build Support

a Pragmatic Approach to the Software Build Process

Atos Origin

Randy Marques

Randy.Marques@atosorigin.com

Technical Automation
In-Product Software

Atos Origin

Introduction
Who am I

- **Randy Marques - CASE Consultant**
  - **Atos Origin Technical Automation (TA)**
    - **In-Product Software (IPS)**
  - **Nederlands Normalisatie Instituut (NEN)**
    - **Nederlandse Programmeertalen Commissie (NC 381 22)**
      - **WG14 (ANSI-C)**
      - **JTC1 SC22WG14 (ANSI-C)**
  - **Software Engineering since 1971**
  - **Coding Standards since 1978**
  - **Build Support since 1980**
  - **C Programming since 1983**
  - **Static Analysis since 1993**
  - **Les Hatton's Safer C© trainer since 2001**

Atos Origin

Scope

- Low, Low Level Configuration Management
- Build Support
  - Compiling, Linking
- Issues
  - Directory Structure
  - ‘make’
  - Multi Platform Variants
  - Functionality Variants
  - 3rd party software, Subcontracting, ‘re-use’
  - Releasing

Atos Origin

Overview

- Task Lifecycle
- Problem Areas
- The Solution
- Does it Work
- Final Remarks & Questions

Atos Origin

Getting the Terminology Straight: Task Lifecycle

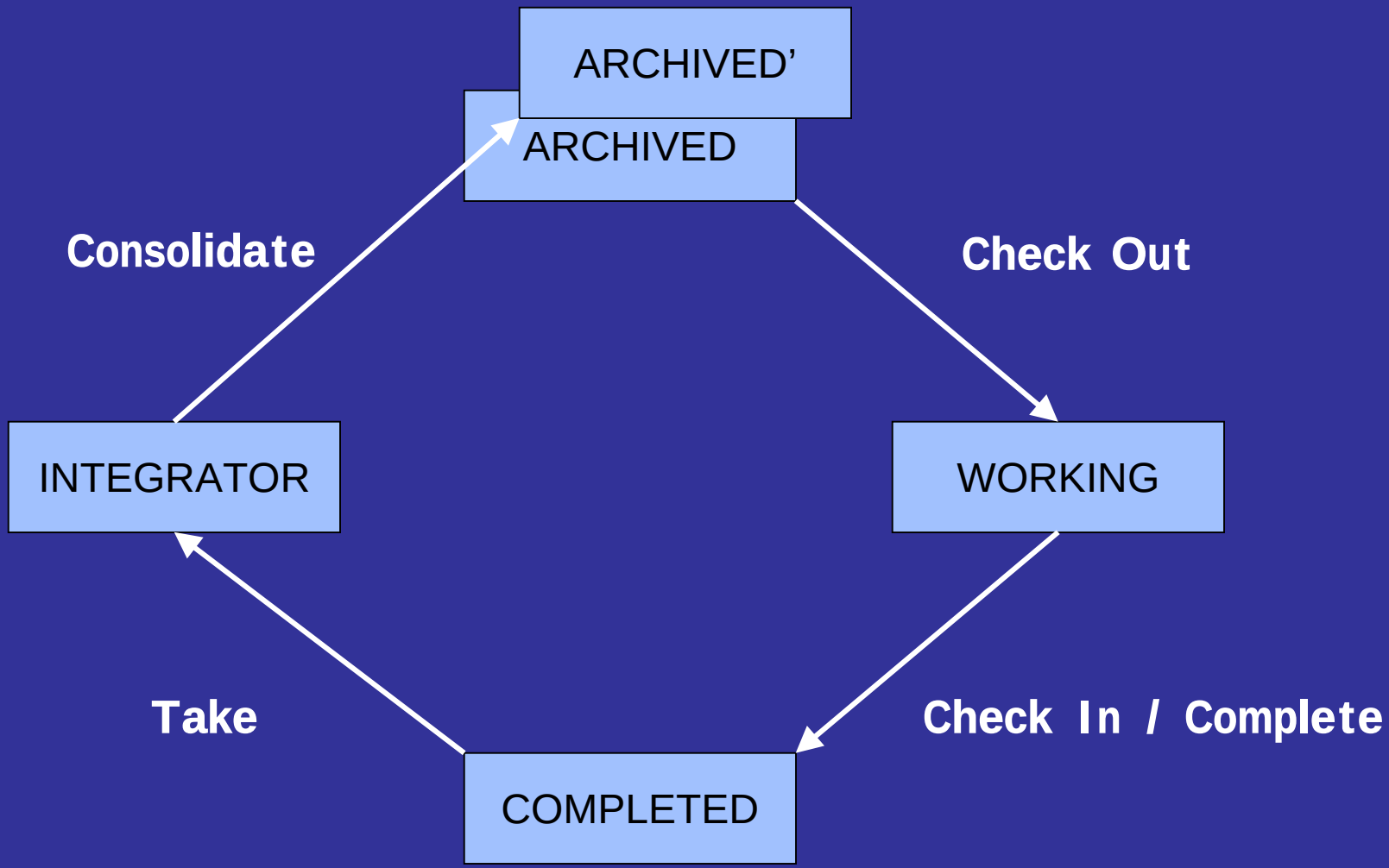
- **Single Level Promotion Model**

- **States:**
  - **ARCHIVED**
    - **Check out**
  - **WORKING**
    - **Complete**
  - **COMPLETED**
    - **Take**
  - **INTEGRATION**
    - **Consolidate**
  - **ARCHIVED'**

- **States:**
  - **ARCHIVED**
    - **Check out**
  - **WORKING**
    - **Complete**
  - **COMPLETED**
    - **Take**
  - **INTEGRATION**
    - **Reject**
  - **REJECTED**
    - **Accept Reject**
  - **WORKING'**

Atos Origin

Task Lifecycle



Atos Origin

Overview

- Task Lifecycle
- Problem Areas
  - Volume
  - 'make' Files
  - Directory Structure
  - Integrity
  - Variants
  - Tooling
- The Solution
- Final Remarks & Questions

Atos Origin

- **Software Generation:**
  - **Lines of code**
    - **20 files of 100 lines -> 5000 files of 400 lines**
    - **2000 lines -> 2.000.000 lines**
      - **Generation takes better part of the night**
      - **Quick Fix???**
  - **People**
    - **4 programmers -> 80 programmers**
    - **1 Programmer**
      - **1 Check-in every week**
      - **1 build-error every 15 check-ins**
        - Every 75 days 1 build-error
    - **75 Programmers**
      - **Every day the build fails!**

Atos Origin

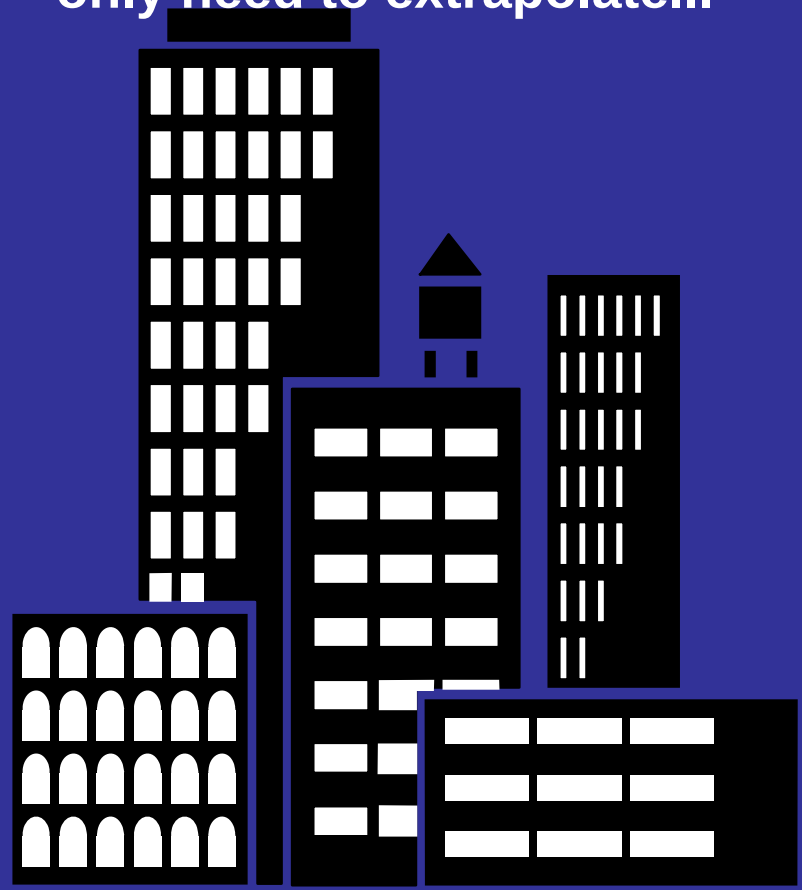
- **Success breeds complexity**
  - **Brian A. White (Software Configuration Management Strategies)**
- **The size of code in consumer electronic products currently doubles every 18 months**
  - **Line-Scan TVs have ~ 500.000 lines of code**
- **Murphy's Law:**
  - **If something CAN go wrong, if you do it often enough, it WILL go wrong**

Atos Origin

From Barn to Skyscraper

- We started off building yard barns

- And we are convinced that to build skyscrapers we only need to extrapolate...



Atos Origin

Overview

- Task Lifecycle
- Problem Areas
  - Volume
  - 'make' Files
  - Directory Structure
  - Integrity
  - Variants
  - Tooling
- The Solution
- Does it Work
- Final Remarks & Questions

Atos Origin

Problem Area: 'make'-Files - 1

- **The essence of ‘make’**
  - **re-generate as little as possible**
  - **Dependencies**
  - **‘A Program for Directing Recompilation’ (GNU-make)**
- **Pollution of ‘make’**
  - **Header-file directory specification**
  - **Compile options**
  - **Generation of various functionality/platform variants**

Atos Origin

Problem Area: 'make'-Files - 2

- **Problems:**
  - **Maintaining Dependencies**
  - **Start-off small & simple. Grow and become too complex**
    - **No-one dares to make modifications**
    - **Not-maintainable make-files (> 1600 lines!)**
  - **Multiple checkout on make-files**
  - **Completely platform/environment dependent**

  - **No Design, No Standards, No QAC, No Reviews, No Fagan-inspections No-one feels responsible**

Atos Origin

Problem Area: 'make'-Files - 3

- **make-File:**
  - Collection of make-rules
  - Macros & Control-statements

- **make-Rule:**

```
target : dependencies
<TAB>  command [; …]
```

```
edit.exe : main.o kbd.o command.o
<TAB>    lnk -o edit.exe main.o \
                 kbd.o command.o
main.o : main.c defs.h
<TAB>    cc -c main.c
kbd.o : kbd.c defs.h command.h
<TAB>    cc -c kbd.c
command.o : command.c defs.h \
                 command.h
<TAB>    cc -c command.c
```

Atos Origin

Overview

- Task Lifecycle
- Problem Areas
  - Volume
  - 'make' Files
  - Directory Structure
  - Integrity
  - Variants
  - Tooling
- The Solution
- Does it Work
- Final Remarks & Questions

Atos Origin

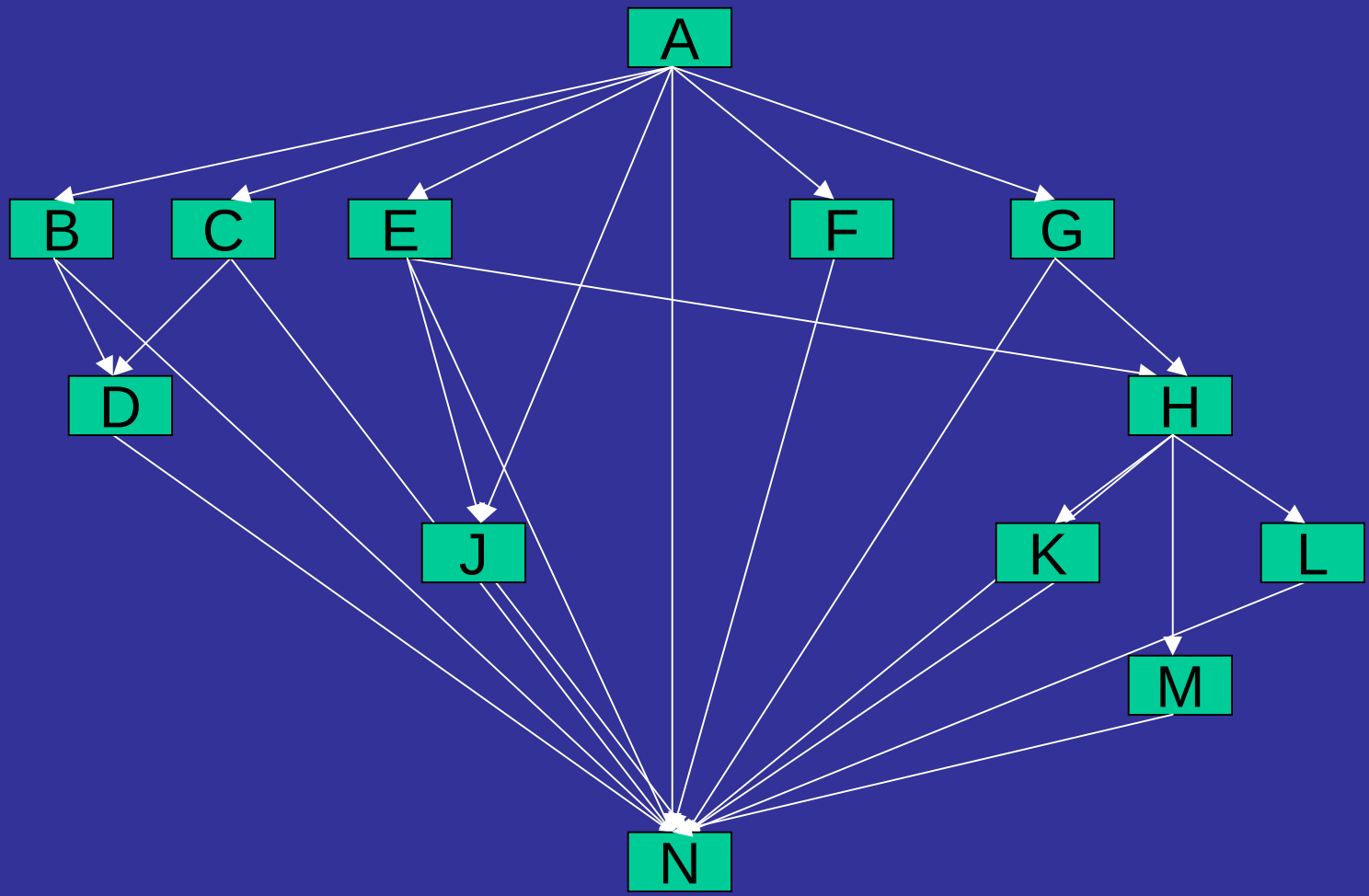
Problem Area: Directory Structure

- **Trying to map a software-structure on a directory structure**
  - **Software Structure:**
    - **Tree-like *Network* Structure**
  - **Directory Structure:**
    - **Tree Structure**

- **Relocating a directory inside the tree structure causes major problems in generation ('make') files**

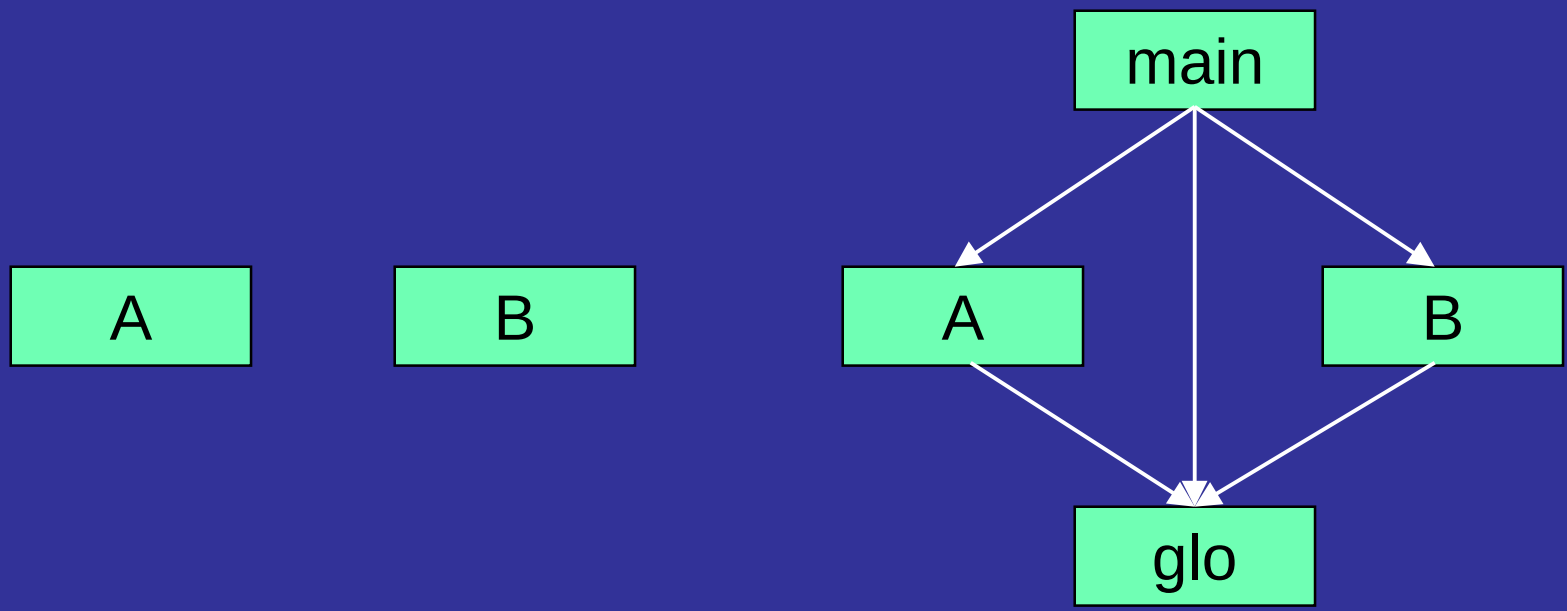
Atos Origin

Software Structure



Component Structure

- **A multi-component system always consists of al least 4 components:**



Atos Origin

Directory Structure - Flexibility

- What effort does it cost to add a new component
  - or compiler, or variant, or platform
- What are the consequences for a SCM-tool
- What do I do with ‘re-use’ from another project

- Do I spend 4 man-months to define my dedicated directory-structure and 'make'-method every time I start a new project?

Atos Origin

Overview

- Task Lifecycle
- Problem Areas
  - Volume
  - 'make' Files
  - Directory Structure
  - Integrity
  - Variants
  - Tooling
- The Solution
- Does it Work
- Final Remarks & Questions

Atos Origin

Problem Area: Integrity

- **How safe/complete is my generation process**
  - **Was really everything compiled before the link?**
  - **Did all files compile properly before Freeze/Archiving/Release?**
- **Given an executable, can I recreate exactly the environment leading to it's creation?**
  - **Sources**
  - **Parameters (compile-time options)**
- **Given the amount of money spent on SCM-tools, how reliable is my ability to regenerate the software?**

Atos Origin

Overview

- Task Lifecycle
- Problem Areas
  - Volume
  - 'make' Files
  - Directory Structure
  - Integrity
  - Variants
  - Tooling
- The Solution
- Does it Work
- Final Remarks & Questions

Atos Origin

Problem Area: Variants - 1

- **Versions**
  - **Taken care of by CMS**
- **Variants / Diversity**
  - **Platform**
  - **Functionality**
    - **Hardware**
    - **Software**
- **Compile-time Options & Conditional Compiles**
- **Many Options**
- **Many Possible Values**

```
#ifdef __SUN__
..
..
..
#endif
```

Atos Origin

Problem Area: Variants - 2

- **Problems:**
  - **Too many permutations**
    - **Always starts with just a few...**
    - **Which combinations are valid?**
  - **Which variant is now in my object-library?**
  - **Which variant did I build my executable with?**
  - **Compile archived version with different option:**
    - **compilation fails**
  - **Not Testable**
  - **Given an executable that was built on a specific date and time:**
    What were the settings that led to this executable?
    (I mean: ALL the settings...)

Atos Origin

Overview

- Task Lifecycle
- Problem Areas
  - Volume
  - 'make' Files
  - Directory Structure
  - Integrity
  - Variants
  - Tooling
- The Solution
- Does it Work
- Final Remarks & Questions

Atos Origin

Problem Area: Additional Tooling

- **How to implement additional tooling**
- **Multiplatform?**
- **Example:**
  - **QAC**
    - **How do I specify which files to check**
    - **How do I obtain header-file include directories (-i)?**
    - **How do I obtain compile time options (-d)?**
    - **‘make’ ?**

Atos Origin

Overview

- Task Lifecycle
- Problem Areas
- The Solution
- Does it Work
- Final Remarks & Questions

• Generic Build Support

- A Concept
- A Set of Agreements
- A Rigid Directory Structure
- No built-in Knowledge of Application
- Generated 'make' Files

- Implementation Support:
  - Scripts
  - Currencies
    - Current System, Current SubSystem, Current Component, Current Target

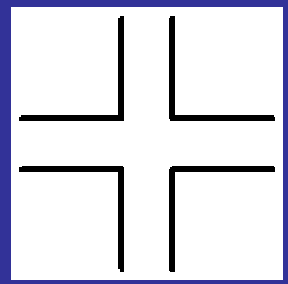
Atos Origin

Solution: Basic Rules

1. Simple approach. No Black Box. KISS.
2. Rigid where no flexibility is required
   Full flexibility where flexibility is needed
3. Do not cater for things we agree to be ‘bad practice’
4. Automate anything that can go wrong when done manually
   (but do not try to achieve 100% automation when in conflict with Rule 1)
5. Must be generic (no knowledge of application)
6. Must be target-platform independent
7. Must have a relocatable directory-structure
8. Keep control: never 'push', always 'pull'.

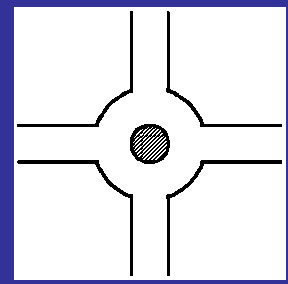
Process Improvement by Conceptual Revolution

- **Cross Roads**



  - Yield right
  - Traffic lights
  - Traffic lights with traffic detection
  - Traffic lights with traffic detection and priority system
  - Traffic lights with traffic detection and priority system with environmental parameters
  - Traffic lights with traffic detection and priority system with environmental parameters enhanced with...

- **Roundabout**



  - Yield left

Atos Origin

Prepare Yourself

- Take a deep breath
- Stretch yourself
- Empty your mind
  - Do NOT think of your current implementation
- Relax
- Lay Back
- Enjoy the show

Atos Origin

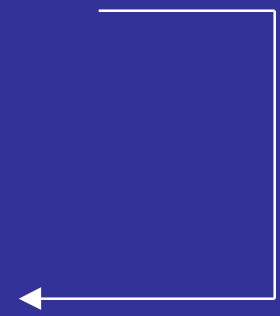
Overview

- Task Lifecycle
- Problem Areas
- The Solution
  - Directory Structure and ‘make’
  - Integrity
  - Variants
  - Finalizing
- Does it Work
- Final Remarks & Questions

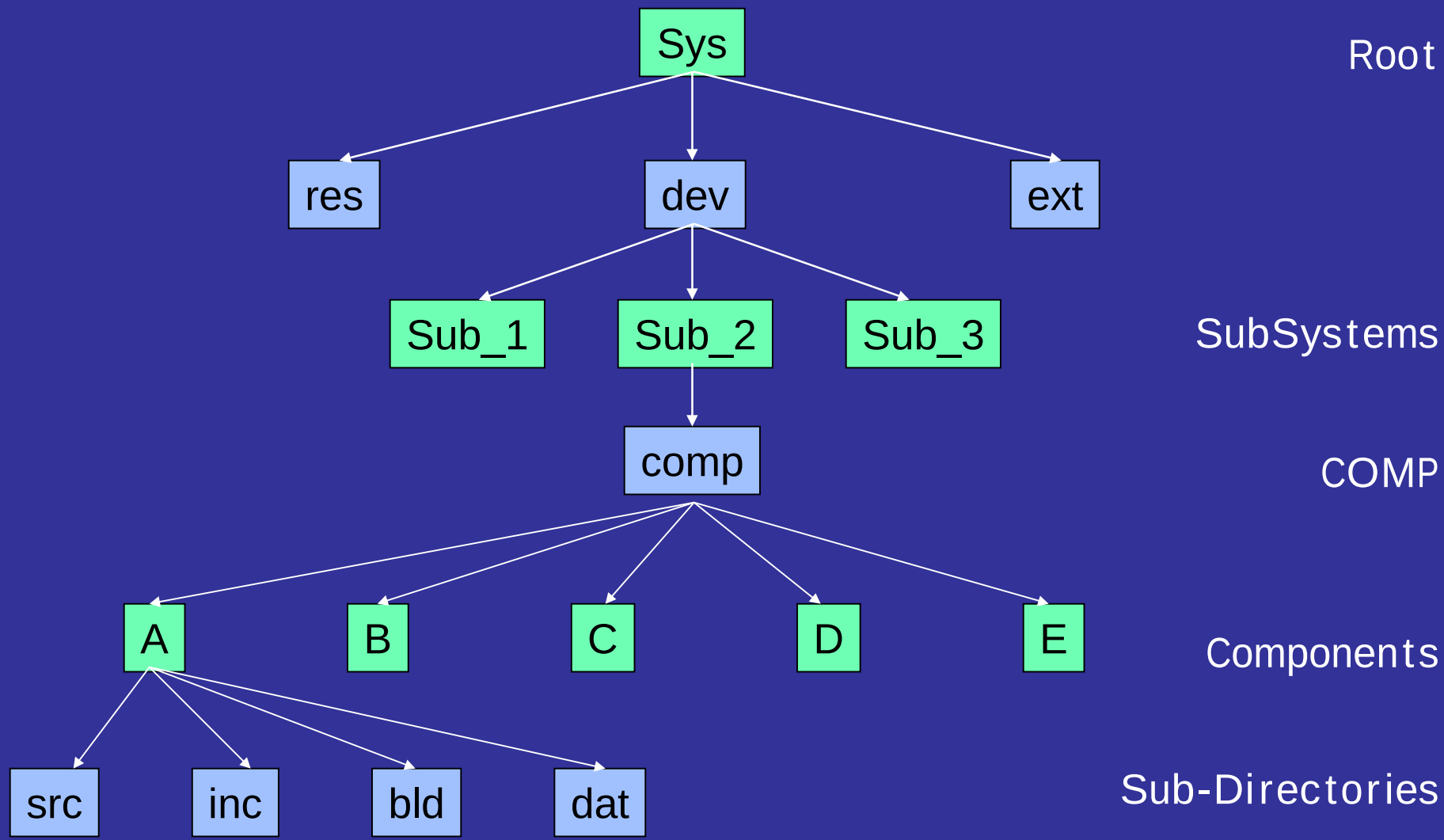
Atos Origin

‘Flat’ Directory Structure

- **Root (System) Directory**
  - **EXT (externals) Directory**
    - **3rd party SW Directories**
  - **DEV (Development) Directory**
    - **SubSystem Directories**
  - **RES (Results) Directory**
    - **SubSystems Transfer Directories**
- **SubSystem Directory**
  - **COMP-Directory**
    - **Component Directories**
      - **Sub-Directories (SRC, INC, BLD, etc)**



‘Flat’ Directory Structure



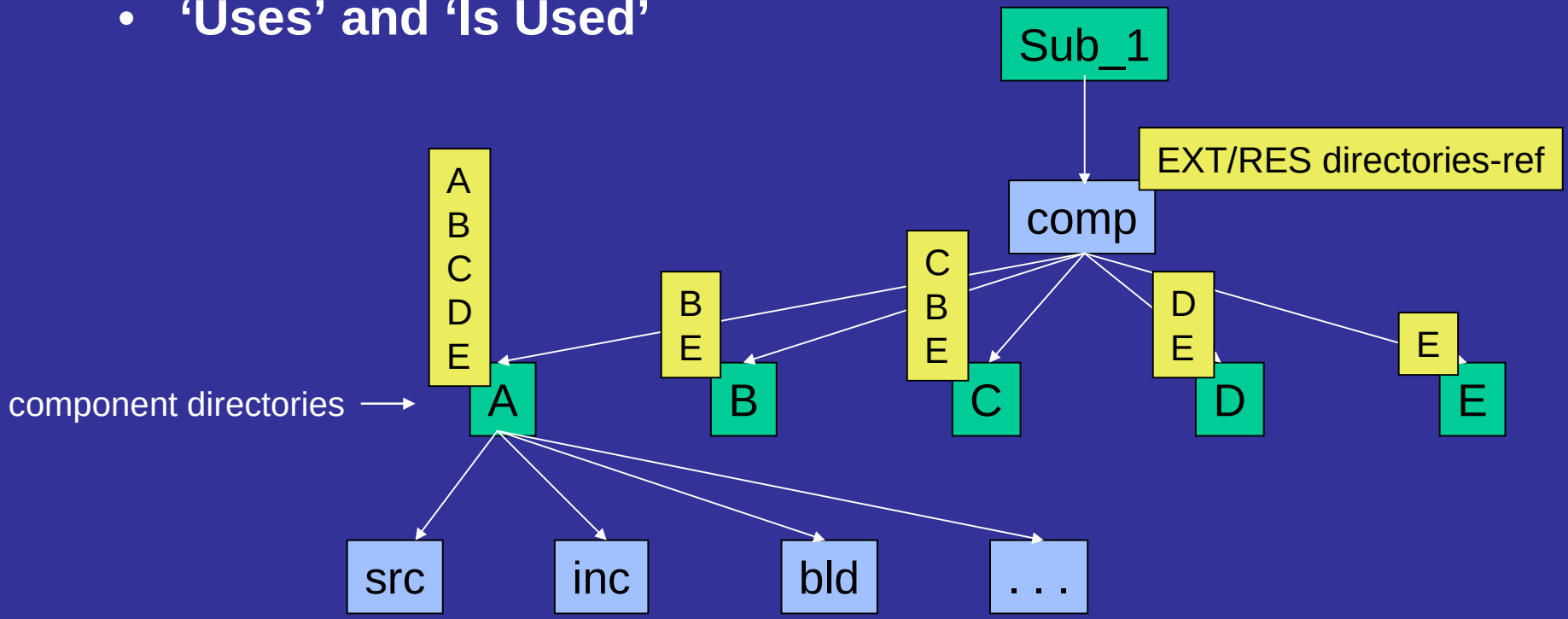
Atos Origin

Sub Directories

- SRC
  - Sources
- INC
  - Global (exported) Header-files
- LOC
  - Local Header-files
- BLD
  - Contains Target BLD-Directories
    - Results of generation (compilations)
- more...

Scoping

- **Scope-files contain component-names, no directory specs.**
- **SCOPE.GBS in Component Directory**
  - **Specifies the 'VIEW'**
- **'Uses' and 'Is Used'**



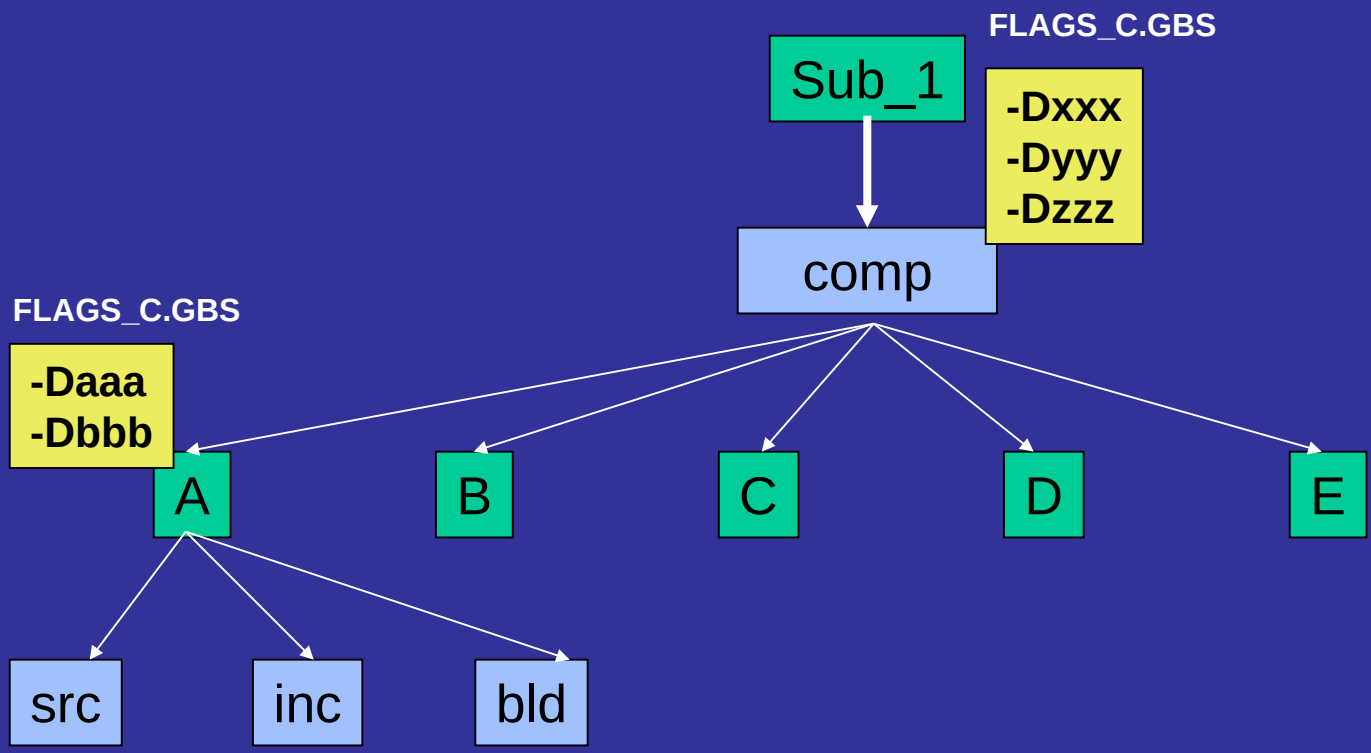
Atos Origin

Compile-time Options - 1

- **Options are placed separately in option-files**
  - **Options for all C-files in project:**
    - **FLAGS_C.GBS In COMP directory**
  - **Options for all C-files in component:**
    - **In FLAGS_C.GBS in Component directory**

- **For compilation, options are placed in de order specified above.**
  - **Last option wins...**

Compile-Time Options - 2



Atos Origin

Generating a Compilation

- **Given a source file of the current Component of the current SubSystem of the current System with a current Target, we have:**
  - **Source File name (file.c)**
  - **Compiler to be used**
  - **Extension of object-file name (.o)**
  - **Object-file name (file.o)**
  - **Header-File Directory information**
  - **Compile Options Information**
  - **Input & Output Directory**
- **So we can have a generic script that generates a dedicated compile command.**

Atos Origin

Generating a Link

- **Traditionally done in ‘make’ file**

- **Enter: Link-file**
  - **Works the same way as 'compile file'**
  - **name.glk => name.exe**
  - **Contains:**

    <component>:<objectfile-name>
  - **Also:**

    #include ...
  - **So we can have a generic script that generates a dedicated link command.**

- `Name.glk => Name.exe`
- `Contents:`

```
A:a.o
A:a1.o
B:b.o
B:b1.o
C:c.lib
```

Atos Origin

Generating a Library

- **Also traditionally done in ‘make’ file**

- **Enter: Library-file**
  - **Works the same way as 'compile file'**
  - **name.glb => name.lib**
  - **Contains:**

    <component>:<objectfile-name>
  - **Also:**

    #include …
  - **So we can have a generic script that generates a dedicated archive command.**

- `Name.glb => Name.lib`
- `Contents:`

```
A:a.o
A:a1.o
B:b.o
B:b1.o
```

Atos Origin

Generating a SubSystem

- In the SRC directories of all Components:
  - Compile all *.c files into objects
- In the SRC directories of all Components:
  - Archive all *.glb files into libraries
- In the SRC directories of all Components:
  - Link all *.glk files into executables
- All results go to the current BLD/<target> directories

Atos Origin

Flavors of 'generate'

- generate `file.c`
- generate <list of individual files from same or other component>
- generate <all files in one or more components>
- generate <all files in one or more subsystems>
- etc...

Atos Origin

Generating a make-file

- **For each Component:**
  - **Take each file in the SRC-directory as a 'make'-target**
  - **Parse the file using the same information as for generation of Compile, Archive or Linking to determine dependencies**
  - **Generate the make-file**

Atos Origin

Flavors of ‘make’

- make `image.exe`
- make <list of specific files>
- make <all files in list of specific components>
- make <SubSystem>
- etc...

Atos Origin

Differences between 'generate' and 'make'

- 'generate'
  - you specify the source (e.g. file.c)
  - only the specified file(s) will be generated
  - all the specified files will be generated
- 'make'
  - you specify the resulting file (e.g. file.o)
  - other files (even in other components) may be generated
  - specified files may or may not be (re-)generated

Atos Origin

Overview

- Task Lifecycle
- Problem Areas
- The Solution
  - Directory Structure and ‘make’
  - Integrity
  - Variants
  - Finalizing
- Final Remarks & Questions

Atos Origin

Verifying Builds

- **When the generation of a file fails the possibly generated file in the BLD will be deleted.**
  - **In case of compilation or archiving, subsequent steps (e.g. link) will also fail**

- **Before CONSOLIDATION we can now check whether a file generated properly:**

  If there is no corresponding file in the BLD directory, the CONSOLIDATION may NOT be executed.

```
SRC:<name>.<type>
    must match
BLD:<name>.*
```

Atos Origin

Overview

- Task Lifecycle
- Problem Areas
- The Solution
  - Directory Structure and ‘make’
  - Integrity
  - Variants
  - Finalising
- Final Remarks & Questions

Atos Origin

Working With Variants

- **Versions**
  - **Taken care of by SCM**
- **Variants**
  - **Platform**
  - **Functionality**
    - **Hardware**
    - **Software**

Atos Origin

Platform Variants

- Conditional Compiles
- Multiple Sources (Link-time diversity)
  - file.c
    - file_pc.c => file_pc.o time()
    - file_vms.c => file_vms.obj time()
    - file_sun.c => file_sun.o time()
  - BLD
    - PC <= Target Directory
    - VMS
    - SUN
- Combinations
- Platform dependent functionality should be isolated into one or two components

Atos Origin

Functionality Variants - 1

- **Multiple Sources (Link-Time diversity)**
- **Configuration Files**

Atos Origin

Functionality Variants - 2

- **Traditional Approach:**

```
MAKE-FILE:
   -D RECORDER

FILE.C:
   #ifdef RECORDER
      …
      …
    #else
      …
      …
    #endif
```

Atos Origin

Functionality Variants - 3

```
CFG1.C:
    bool recorder( void)
        { return TRUE; }
```

```
CFG2.C:
    bool recorder( void)
        { return FALSE; }
```

```
FILE.C:
   if (recorder())
   {
        ...
        do_recorder();
    } else
    {
        ...
        ...
    }
```

- **Configuration Files:**
  - Link FILE.O either with CFG1.O or CFG2.O
  - Need not be static
    - Read a file (.ini)
    - Read Hardware Memory (jumpers)
    - Ask user

Atos Origin

Overview

- Task Lifecycle
- Problem Areas
- The Solution
  - Directory Structure and ‘make’
  - Integrity
  - Variants
  - Finalizing
- Final Remarks & Questions

Atos Origin

Subdirectories - 1

- SRC
  - Trigger in generation process. Results go to BLD directory
  - Scanned for dependencies.
  - Archived in SCM
- INC
  - Exported to generation of other components
  - Scanned for dependencies.
  - Archived in SCM
- LOC
  - Scanned for dependencies.
  - Archived in SCM
- BLD
  - <target> Contains results of SRC generations
  - Exported to generation of other components
  - Scanned for dependencies
  - Full delete of contents allowed

Atos Origin

Subdirectories - 2

- **DAT**
  - **Contains any other data that partakes in the generation of the resulting delivery. e.g.:**
    - **bitmaps**
    - **scripts**
  - **Archived in SCM**
- **SAV**
  - **Contains anything that has to be kept but does not partake in the generation-process, at this time**
    - **old stuff**
  - **Archived in SCM**

Atos Origin

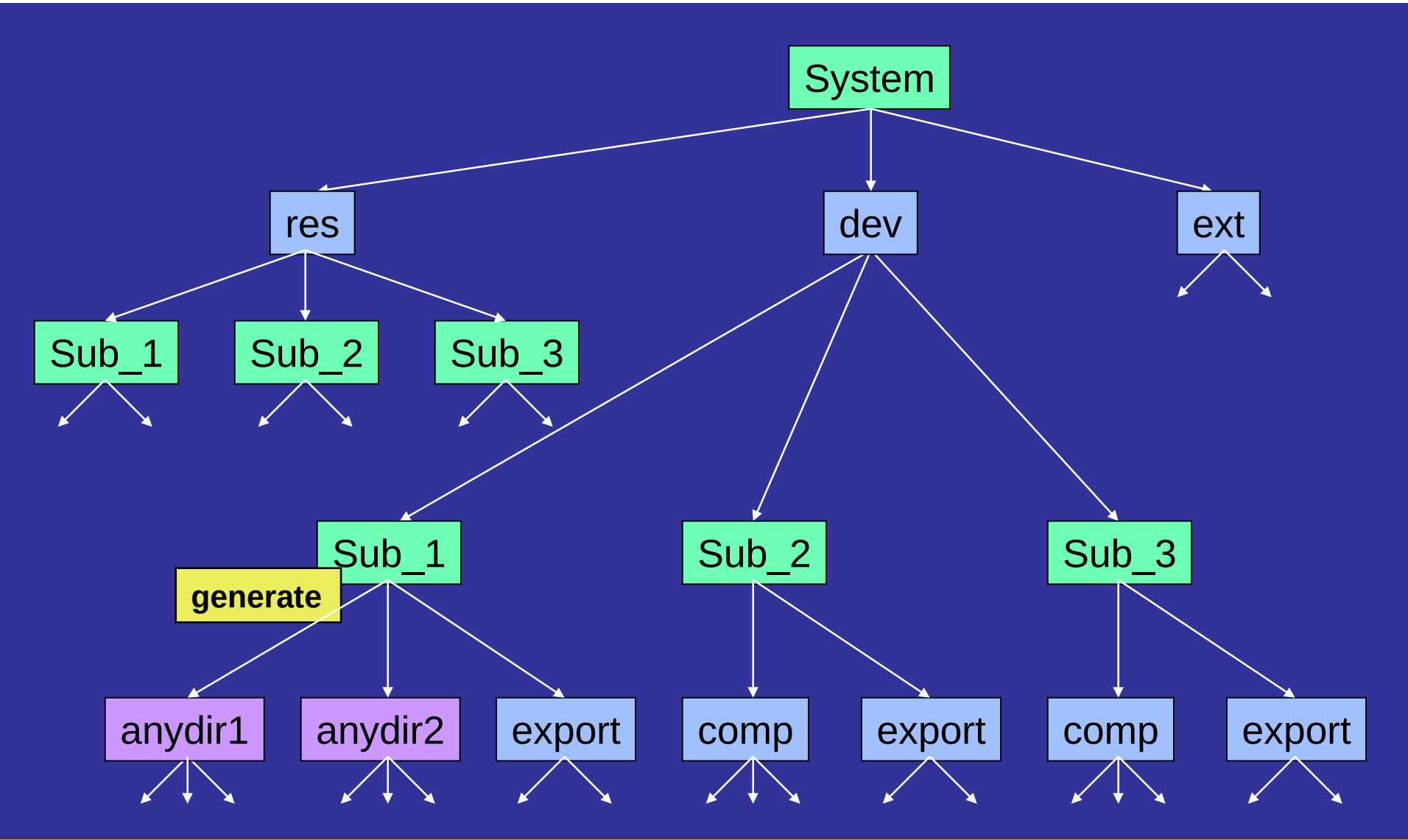
The Bigger Picture - 1

- **Handling of SubSystems on a System Level**
  - **export directory**
    - **deliverables**
    - **export.gbs file in every component**
    - **'fillexport'**
  - **res directory**
    - **'fillres'**
- **Generation of non-compliant SubSystems**
  - **export directory**
  - **'generate' script**

Direct reference from one SubSystem to another is absolutely not allowed

Atos Origin

The Bigger Picture - 2



Atos Origin

The Bigger Picture - 3

- **To generate a whole System:**
  - **Generate first SubSystem**
    - **GBS-compliant:**
      - **'generate' or 'make' the SubSystem**
      - **'fillexport' (copies files to 'export' directory)**
    - **Non GBS-compliant:**
      - **Execute 'generate' script in SubSystem directory**
        At the end, deliverables must be in 'export' directory
  - **Make deliverables available to other SubSystems by running 'fillres'. 'fillres copies the contents of the 'export' directory to the 'res' directory**
  - **Generate next SubSystem**
    - **etc**

Atos Origin

Overview

- Task Lifecycle
- Problem Areas
- The Solution
- Does it Work
- Final Remarks & Questions

Atos Origin

Does it Work - 1

- **GBS is based on experiences with 'CCS', the Code Control System developed at Philips Medical Systems in 1985**
  - **Designed and developed by the author**
  - **Used for over 15 years by a variety of small and large projects**
    - **still in use for maintenance projects**
  - **Written for VAX/VMS in DCL**
  - **1 Dissatisfied user: Tried to use CCS in a 'non-intended way'**
- **Now in development for UNIX and Windows-NT**
  - **Written in Perl**

Atos Origin

Does it Work - 2

- **Pilot project at Philips ASA-Lab - 1**
  - **MultiSite (Eindhoven, Dublin, Suresnes and Bangalore)**
  - **Approx. 30 developers (UNIX + NT)**
  - **Processors: ST20, MIPS5.1**
  - **Compilers: OpenTv (1), C (2), C++ (2), Assembler(2)**
  - **Application: Multi Language variants: 5 x 2 Executables**
  - **Size**
    - **13 SubSystems**
    - **241 Components**
    - **1182 Source-files**
    - **611 Local header files**
    - **545 Global header files**
    - **160 Libraries**
    - **41 Executables**
    - **8 Externals (3rd party components)**

Atos Origin

Does it Work - 3

- **Pilot project at Philips ASA-Lab - 2**
  - **Originally:**
    - **Both platforms: existing code in separate archives**
    - **Considerable amount of shared code**
      - **Good idea: Universal Archive (let same code go through 2 compilers)**
    - **Considered major effort:**
      - **6 man-months in 3 months**
      - **Non-workable situation during 1 month for 10 people**
    - **2x5x4=40 executables from 20 full generations**
  - **With introduction of GBS:**
    - **First platform: silently**
    - **Second platform:**
      - **1 man-month in 3 weeks**
      - **Same way of working for all platforms:**
        - No impact for others during introduction
    - **2x5=10 executables from 2 full generations**

Atos Origin

Does it Work - 4

- **Pilot project at Philips ASA-Lab - 3**
  - **Originally:**
    - **Developers:**
      - **Had no idea of how the build (make) process worked.**
      - **Did not know how to introduce new (source-)files**
      - **Did not know where to find included headerfiles and/or libraries**
      - **Did not know how to generate a new library or executable**
      - **Did not know how to generate another SubSystem**
    - **The 'make' process was unsafe**
  - **With introduction of GBS:**
    - **Developers:**
      - **Know exactly where to place and find files**
      - **Understand the 'make' process**
      - **Agree: More output per worked hour**
    - **Safe 'make' process**

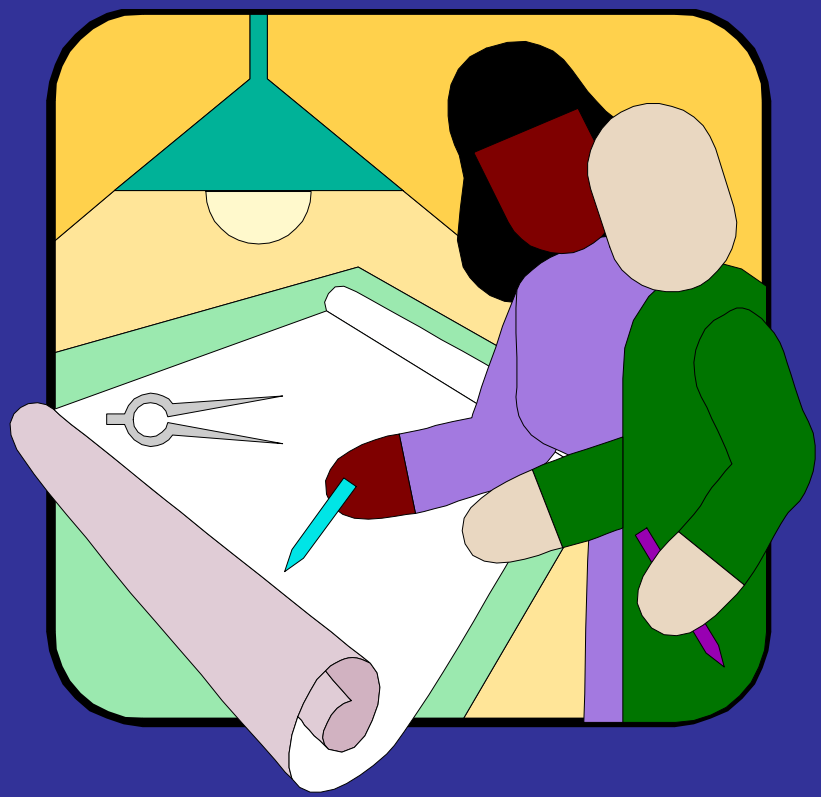
Atos Origin

Overview

- Task Lifecycle
- Problem Areas
- The Solution
- Does it Work
- **Final Remarks & Questions**

Good Design Poor Masonry…

- **Design gets lots of attention. (Architect)**



- **(Too) Little concern / appreciation for the building process.**
  (Construction Supervisor)
  (Dutch: bouwmeester)

Atos Origin

Final Remarks & Questions

- No 100% solution
- Combat proven
- Requires change of focus (traffic lights)

- Questions
  - Task Lifecycle
  - Problem Areas
    - 'make' files
    - Directory Structure
    - Integrity
    - Variants
    - Tooling
  - The Solution
    - Flat and rigid directory structure
    - make-file generation
    - The Bigger Picture
  - Does it Work